

High Performance Embedded Computing Software Initiative (HPEC-SI)

Jeremy Kepner* (kepner@ll.mit.edu) MIT Lincoln Laboratory, Lexington, MA 02420

Abstract

The High Performance Embedded Computing Software Initiative (see www.hpec-si.org) is addressing the military need to advance the state of embedded software development tools, libraries, and methodologies to retain the nation's military technology advantage in increasingly software-based systems. Key accomplishment include completion of the first demonstration and the development of the Parallel VSIPL++ standard. Currently the HPEC-SI effort is on track towards its goal of changing the state-of-the-practice in programming DoD HPEC SIP systems.

1 Introduction

The High Performance Embedded Computing Software Initiative (HPEC-SI) involves a partnership of industry, academia, and government organizations to foster software technology insertion demonstrations, to advance the development of existing standards, and to promote a unified computation/communication embedded software standard. The goal of the initiative is software portability: to enable "write-once/run-anywhere/run-any-size" for applications of high performance embedded computing (see [7, 4, 10, 8, 9, 18, 12]).

This paper gives a brief overview of the HPEC-SI program objectives, technical objectives and program plans. Detailed progress of the demonstration, development and applied research activities that are taking place within the HPEC-SI can be found in the HPEC2002[15, 20, 27], GOMAC2002[26, 5, 11, 21, 23], GOMAC2003[28, 6, 14, 17, 22], and other conferences[16, 13].

*This work is sponsored by the High Performance Computing Modernization Office, under Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

2 Program Objectives

HPEC-SI is organized around demonstrations, standards development and applied research. Each of these activities is overseen by a Working Group. The demonstrations team Prime contractors with FFRDC or academic partners to use currently defined standards, evaluate their performance, and report on how well their needs are being met. The first demonstration was with the Common Imagery Processor (CIP) and successfully showed the use of MPI communication standard ([1]) and the VSIPL computation standard ([2]) to achieve portability (while preserving performance) across shared servers and distributed memory embedded systems. The Development Working Group is extending the VSIPL standard to include parallel object-oriented software practices already prototyped by the research community. This effort is tightly coupled with military demonstrations, and provides the next generation of standards with direct feedback from the military user base. The Applied Research Working Group is also taking a longer term view to assess the potential impact of a variety of emerging technologies such as: fault tolerance and dynamic scheduling, self-optimization, and next generation high productivity languages.

3 Technical Objectives

The HPEC-SI program uses three principal metrics to measure the progress of its efforts:

- Portability (reduction in lines-of-code to change port/scale to new system);
- Productivity (reduction in overall lines-of-code);
- Performance (computation and communication benchmarks).

Traditionally, it has always been possible to improve in two of the above areas while sacrificing the third. HPEC-SI aims to improve quantitatively in all three areas.

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 20 AUG 2004		2. REPORT TYPE N/A		3. DATES COVERED -	
4. TITLE AND SUBTITLE High Performance Embedded Computing Software Initiative (HPEC-SI)				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) MIT Lincoln Laboratory, Lexington, MA 02420				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release, distribution unlimited					
13. SUPPLEMENTARY NOTES See also ADM001694, HPEC-6-Vol 1 ESC-TR-2003-081; High Performance Embedded Computing (HPEC) Workshop (7th)., The original document contains color images.					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 34	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

HPEC-SI expects to achieve at least a 3x reduction in the number code changes necessary to port an application across computing platforms. This improvement will primarily be achieved through the use and enhancement of open software standards (MPI and VSIPL) that will insulate applications from the details of the underlying hardware. An equivalent reduction in code changes will also be seen when porting from one size of platform to another. This will be achieved by the development of a unified computation and communication standard (Parallel VSIPL) which will allow applications to be moved from a computer with N processors to a computer with M processors with minimal code changes.

HPEC-SI expects to achieve a 3x reduction in the total number of lines of code necessary to implement an application. This productivity improvement will be primarily be through the use of higher level object oriented languages (e.g. C++) as well as a unified computation and communication library which will abstract away many of code intensive details of writing a parallel program.

HPEC-SI expects to achieve a 1.5x increase in performance over existing approaches on some computation and communication benchmarks. This is primarily due to an increased level of abstraction which allows the increased use of “early binding” in the application, in the library and in the compiler. [Early binding is the process of building data structures in advance that increase performance at runtime.]

4 Summary

The current achievements of HPEC-SI include the successful utilization of the Vector Signal and Image Processing Library (VSIPL) and the Message Passing Interface to demonstrate a tactical synthetic aperture radar (SAR) code running without modifications and at high performance on parallel embedded, server and cluster systems. HPEC-SI is also creating the first parallel object oriented computation standard by adding these extensions to the VSIPL standard. The parallel VSIPL++ standard will allow high performance parallel signal and image processing applications to take advantage of the increased productivity offered by object oriented program as well as the performance advantages found using advanced expression template technology. The draft object oriented specification and reference code are both available on the HPEC-SI website and are being tested by a

variety of early adopters. Finally, HPEC-SI is evaluating advanced software technologies such as fault tolerance and the use of higher level languages to determine which aspects are ready for future standardization. Combined, all of these efforts are successfully changing the state-of-the-practice in programming DoD HPEC SIP systems. Critical to this effort has been the availability of a wide variety of HPCMO systems (Mercury, Sky, SGI, Compaq, IBM, Linux, and FPGA) that has allowed the testing and demonstration of advanced software technologies for DoD signal and image processing applications.

References

- [1] Message Passing Interface (MPI), <http://www.mpi-forum.org/>
- [2] Vector, Signal, Image Processing Library <http://www.vsipl.org/>
- [3] High Performance Embedded Computing Software Initiative <http://www.hpec-si.org/>
- [4] E. Baranoski, High Performance Embedded Computing Software Initiative (HPEC-SI), DoD HPCMO Signal/Image Processing Forum (SIP 2001), May 15, 2001, Biloxi, MS
- [5] D. Campbell and M. Richards, Object Oriented Extensions to the Vector, Signal, and Image Processing Library (VSIPL) standard, DoD HPCMO Signal/Image Processing Forum (SIP 2002/GOMAC 2002), Mar 13, 2002, Monterey, CA
- [6] D. Campbell C++ and parallelism extensions to the VSIPL standard, GOMACTECH 2003, Apr 4, 2003, Tampa, FL
- [7] C. Holland, Keynote Address, High Performance Embedded Computing Workshop (HPEC 2001), Sep. 25, 2001, Lexington, MA
- [8] C. Holland, The Convergence of HPC and HPEC, Panel at Supercomputing 2001, Nov. 13, Denver, CO
- [9] J. Grosh, Invited Talk: A DoD Perspective on High Performance Computing, High Performance Embedded Computing Workshop (HPEC 2001), Nov. 27, 2001, Lexington, MA

- [10] J. Kepner, Defining Future High Performance Embedded Computing Software, DoD HPCMO Signal/Image Processing Forum (SIP 2001), May 15, 2001, Biloxi, MS
- [11] J. Kepner, High Performance Embedded Computing Software Initiative (HPEC-SI), DoD HPCMO Signal/Image Processing Forum (SIP 2002/GOMAC 2002), Mar 13, 2002, Monterey, CA
- [12] J. Kepner, DoD Sensor Processing: Applications and Supporting Software Technology, Supercomputing 2002, Baltimore, MD, Nov 2002.
- [13] J. Kepner, High Performance Embedded Computing Software Initiative (HPEC-SI), HPCMO User's Group Conference, Jun 10 2003, Bellevue, WA
- [14] J. Kepner, High Performance Embedded Computing Software Initiative (HPEC-SI), GOMACTECH 2003, Apr 4, 2003, Tampa, FL
- [15] M. Mitchell and J. Oldham, VSIPL++: Intuitive Programming Using C++ Templates HPEC 2002, Sep 2002, Lexington, MA
- [16] M. Mitchell and J. Oldham, Parallel VSIPL using VSIPL++, Supercomputing 2002, Baltimore, MD, Nov 2002.
- [17] M. Mitchell and J. Oldham, VSIPL++: Intuitive Programming Using C++ Templates GOMACTECH 2003, Apr 4, 2003, Tampa, FL
- [18] Maj Gen P. Nielsen, Keynote Address: Perspective on Embedded Computing HPEC 2002 Workshop, Lexington, MA, Sep 2002
- [19] E. Rutledge and J. Kepner, PVL An Object Oriented Software Library for Parallel Signal Processing, Cluster 2001, October 9, 2001, Newport Beach, CA
- [20] M. Richards et al, Development Status of the Vector, Signal, and Image Processing Library (VSIPL), HPEC 2002 Workshop, Lexington, MA, Sep 2002
- [21] T. Skjellum and G. Bourdreaux, Early Binding and C++ for VSIPL and VSIPL++, DoD HPCMO Signal/Image Processing Forum (SIP 2002/GOMAC 2002), Mar 13, 2002, Monterey, CA
- [22] T. Skjellum and G. Bourdreaux, Early Binding and C++ for VSIPL and VSIPL++, GOMACTECH 2003, Apr 4, 2003, Tampa, FL
- [23] H. Spaanenburg, Virtualization, DoD HPCMO Signal/Image Processing Forum (SIP 2002/GOMAC 2002), Mar 13, 2002, Monterey, CA
- [24] B. Sroka, Standards-Based Real-Time Embedded High Performance Computing: Common Imagery Processor, DoD HPCMO Signal/Image Processing Forum (SIP 2001), May 15, 2001, Biloxi, MS
- [25] B. Sroka and D. Szakovits, Standards-Based Real-Time Embedded High Performance Computing: Common Imagery Processor, High Performance Embedded Computing Workshop (HPEC 2001), Nov. 29, 2001, Lexington, MA
- [26] B. Sroka, Standards-Based Real-Time Embedded High Performance Computing: Common Imagery Processor, DoD HPCMO Signal/Image Processing Forum (SIP 2002/GOMAC 2002), Mar 13, 2002, Monterey, CA
- [27] B. Sroka and D. Szakovits, Standards-Based Real-Time Embedded High Performance Computing: Common Imagery Processor, HPEC 2002, Sep 2002, Lexington, MA
- [28] B. Sroka, Standards-Based Real-Time Embedded High Performance Computing: Common Imagery Processor, GOMACTECH 2003, Apr 4, 2003, Tampa, FL



High Performance Embedded Computing Software Initiative (HPEC-SI)

Dr. Jeremy Kepner / Lincoln Laboratory

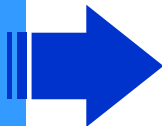
This work is sponsored by the Department of Defense under Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the United States Government.



Outline



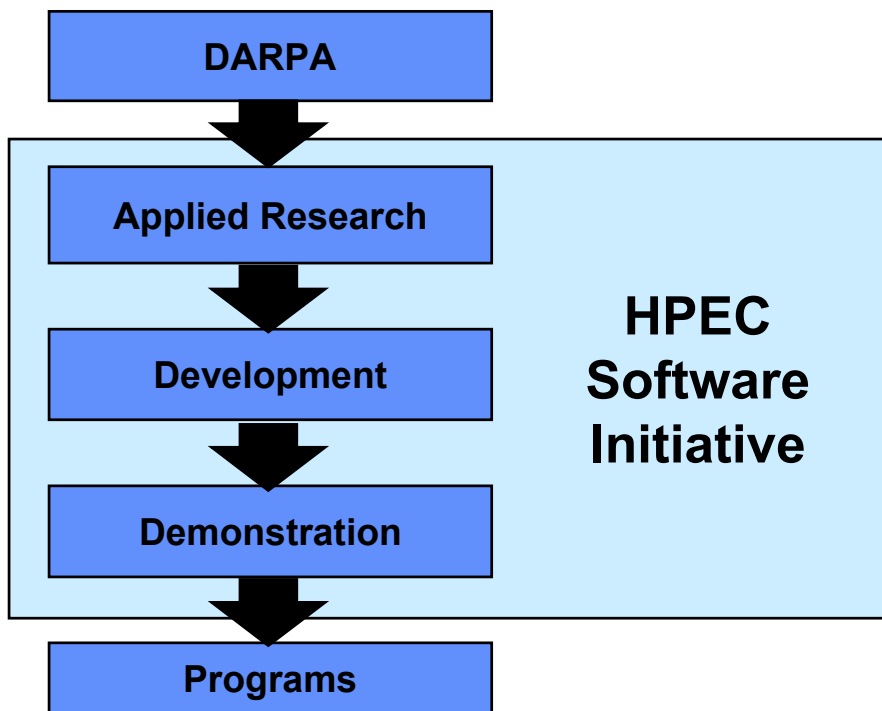
- **Introduction**



- *Goals*
- *Program Structure*

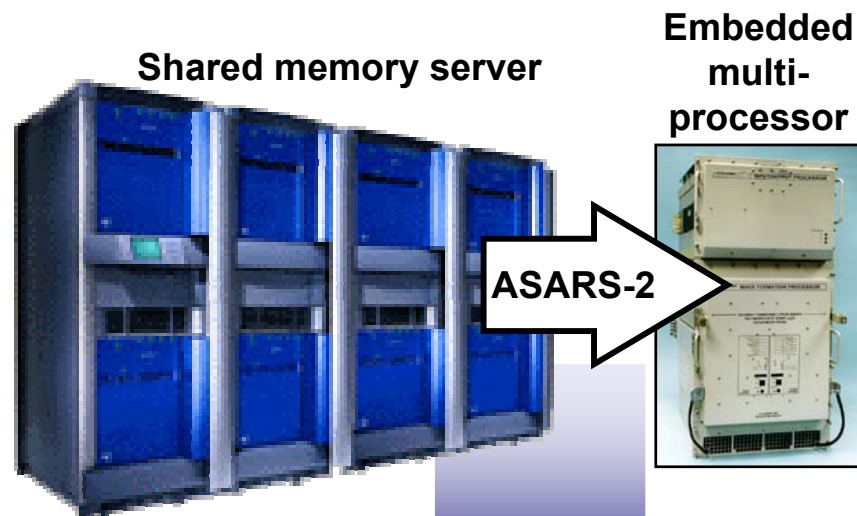
- Demonstration
- Development
- Applied Research
- Future Challenges
- Summary

Overview - High Performance Embedded Computing (HPEC) Initiative



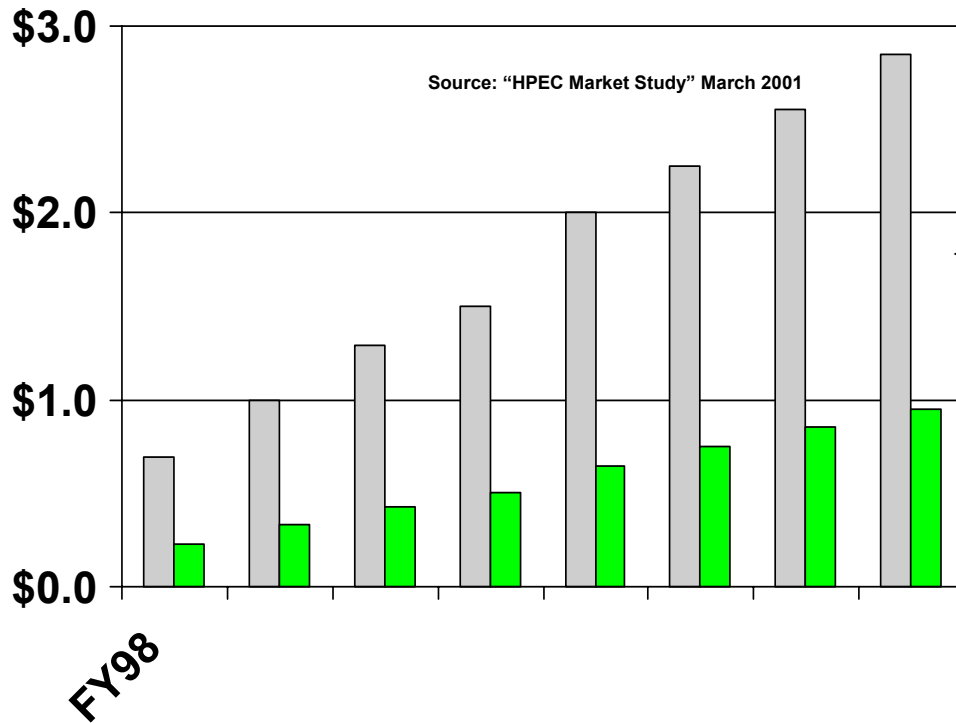
Challenge: Transition advanced software technology and practices into major defense acquisition programs

Common Imagery Processor (CIP)





Why Is DoD Concerned with Embedded Software?



Estimated DoD expenditures for embedded signal and image processing hardware and software (\$B)

- COTS acquisition practices have shifted the burden from “point design” hardware to “point design” software
- Software costs for embedded systems could be reduced by one-third with improved programming models, methodologies, and standards

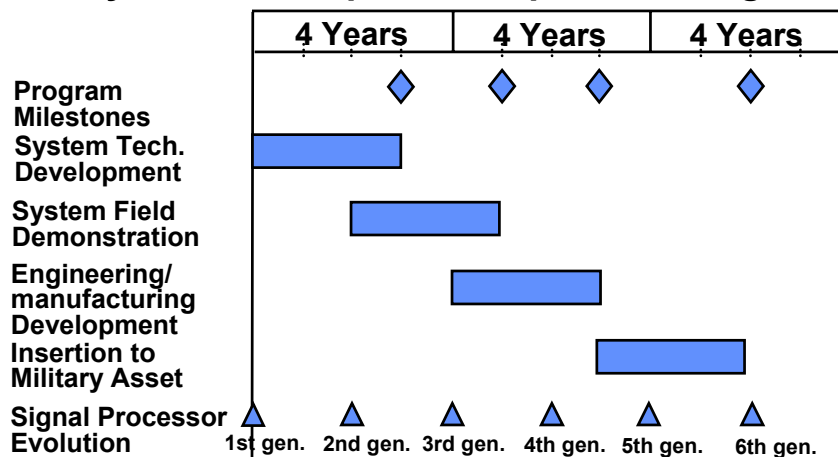
Inadequacy of Software Practices & Standards



- **High Performance Embedded Computing pervasive through DoD applications**

- **Airborne Radar Insertion program**
85% software rewrite for each hardware platform
- **Missile common processor**
Processor board costs < \$100k
Software development costs > \$100M
- **Torpedo upgrade**
Two software re-writes required after changes in hardware design

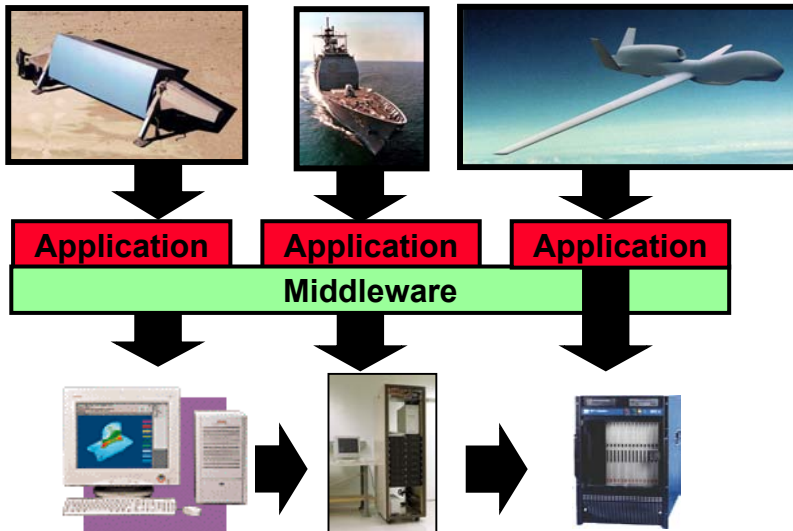
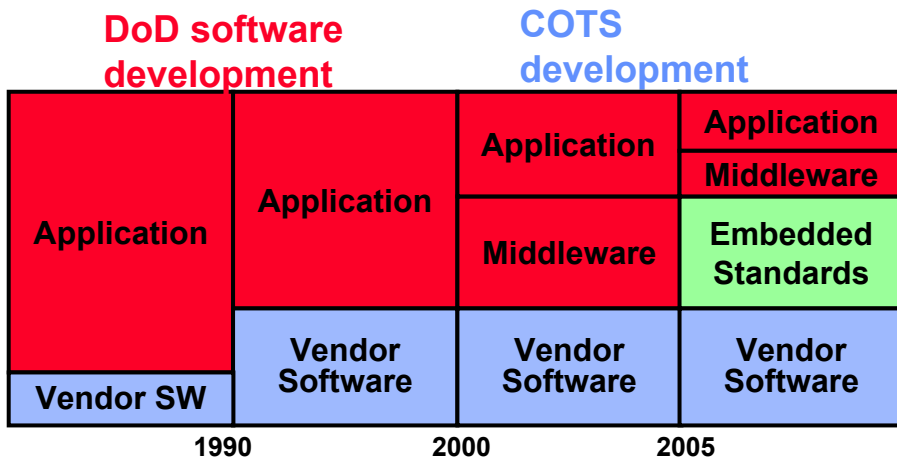
System Development/Acquisition Stages



Today – Embedded Software Is:

- Not portable
- Not scalable
- Difficult to develop
- Expensive to maintain

Evolution of Software Support Towards “Write Once, Run Anywhere/Anysize”



- Application software has traditionally been tied to the hardware
- Many acquisition programs are developing stove-piped middleware “standards”
- Open software standards can provide portability, performance, and productivity benefits
- Support “Write Once, Run Anywhere/Anysize”

Quantitative Goals & Impact

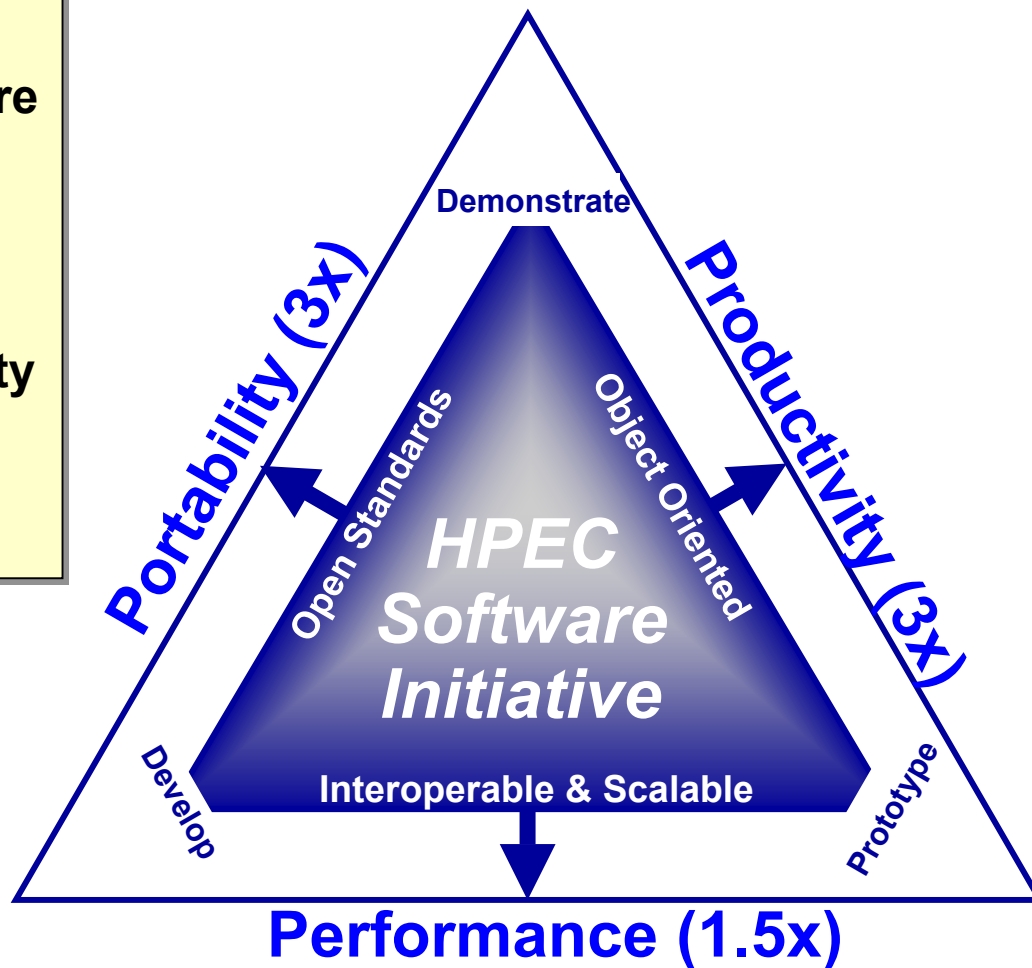
Program Goals

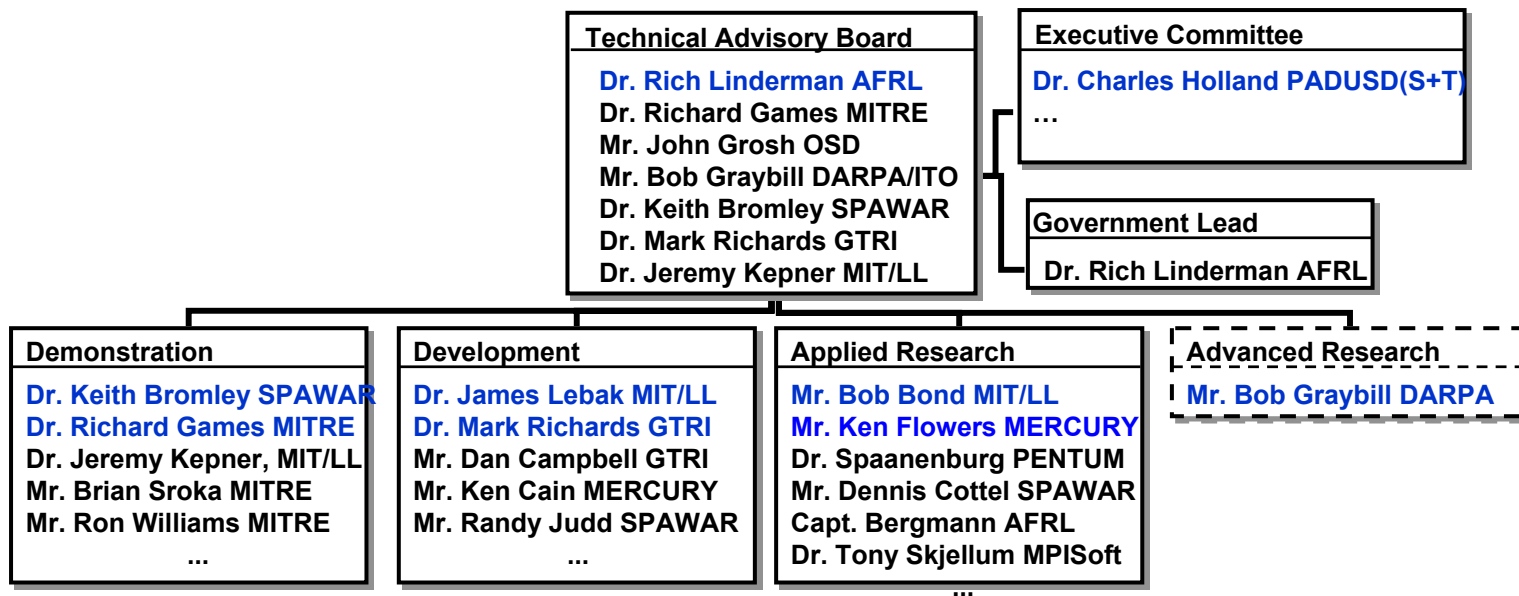
- Develop and integrate software technologies for embedded parallel systems to address portability, productivity, and performance
- Engage acquisition community to promote technology insertion
- **Deliver quantifiable benefits**

Portability: reduction in lines-of-code to change port/scale to new system

Productivity: reduction in overall lines-of-code

Performance: computation and communication benchmarks





- Partnership with ODUSD(S&T), Government Labs, FFRDCs, Universities, Contractors, Vendors and DoD programs
- Over 100 participants from over 20 organizations



HPEC-SI Capability Phases

- First demo successfully completed
- Second demo Selected
- VSIP++ **v0.8** spec completed
- VSIP++ **v0.2** code available
- Parallel VSIP++ **v0.1** spec completed
- High performance C++ demonstrated

Time →

Phase 1

Applied Research:
Unified Comp/Comm Lib

Development:
Object-Oriented Standards

Demonstration:
Existing Standards

VSIP++
MPI

Phase 2

Applied Research:
Fault tolerance

Development:
Unified Comp/Comm Lib

Demonstration:
Object-Oriented Standards

VSIP++

Phase 3

Applied Research:
Hybrid Architectures

Development:
Fault tolerance

Demonstration:
Unified Comp/Comm Lib

Parallel
VSIP++

Functionality ↑

Demonstrate insertions into
fielded systems (CIP)

- Demonstrate 3x portability

High-level code
abstraction (AEGIS)

- Reduce code size 3x

Unified embedded
computation/
communication
standard

- Demonstrate scalability

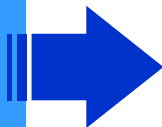


Outline



- Introduction

- **Demonstration**



- *Common Imagery Processor*
- *AEGIS BMD (planned)*

- Development
- Applied Research
- Future Challenges
- Summary

Common Imagery Processor

- Demonstration Overview -

**Common Imagery Processor (CIP)
is a cross-service component**

Sample list of CIP modes

- U-2 (ASARS-2, SYERS)
- **F/A-18 ATARS (EO/IR/APG-73)**
- LO HAE UAV (EO, SAR)
- System Manager

38.5"



CIP*



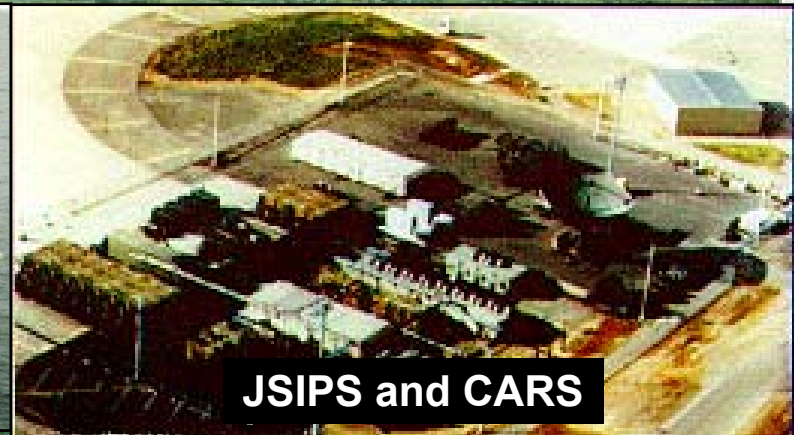
TEG and TES



ETRAC



JSIPS-N and TIS



JSIPS and CARS

Common Imagery Processor

- Demonstration Overview -

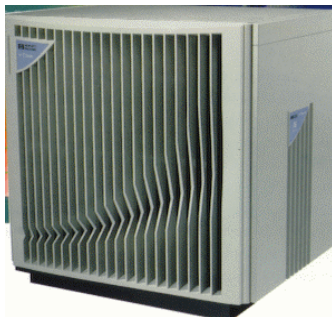
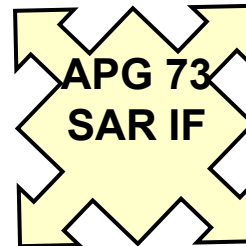


Common Imagery
Processor

- Demonstrate standards-based platform-independent CIP processing (ASARS-2)
- Assess performance of current COTS portability standards (MPI, VSIPL)
- Validate SW development productivity of emerging Data Reorganization Interface
- MITRE and Northrop Grumman



Embedded
Multicomputers



Shared-Memory Servers

Single code base
optimized for all high
performance architectures
provides future *flexibility*



Commodity Clusters
Massively Parallel Processors

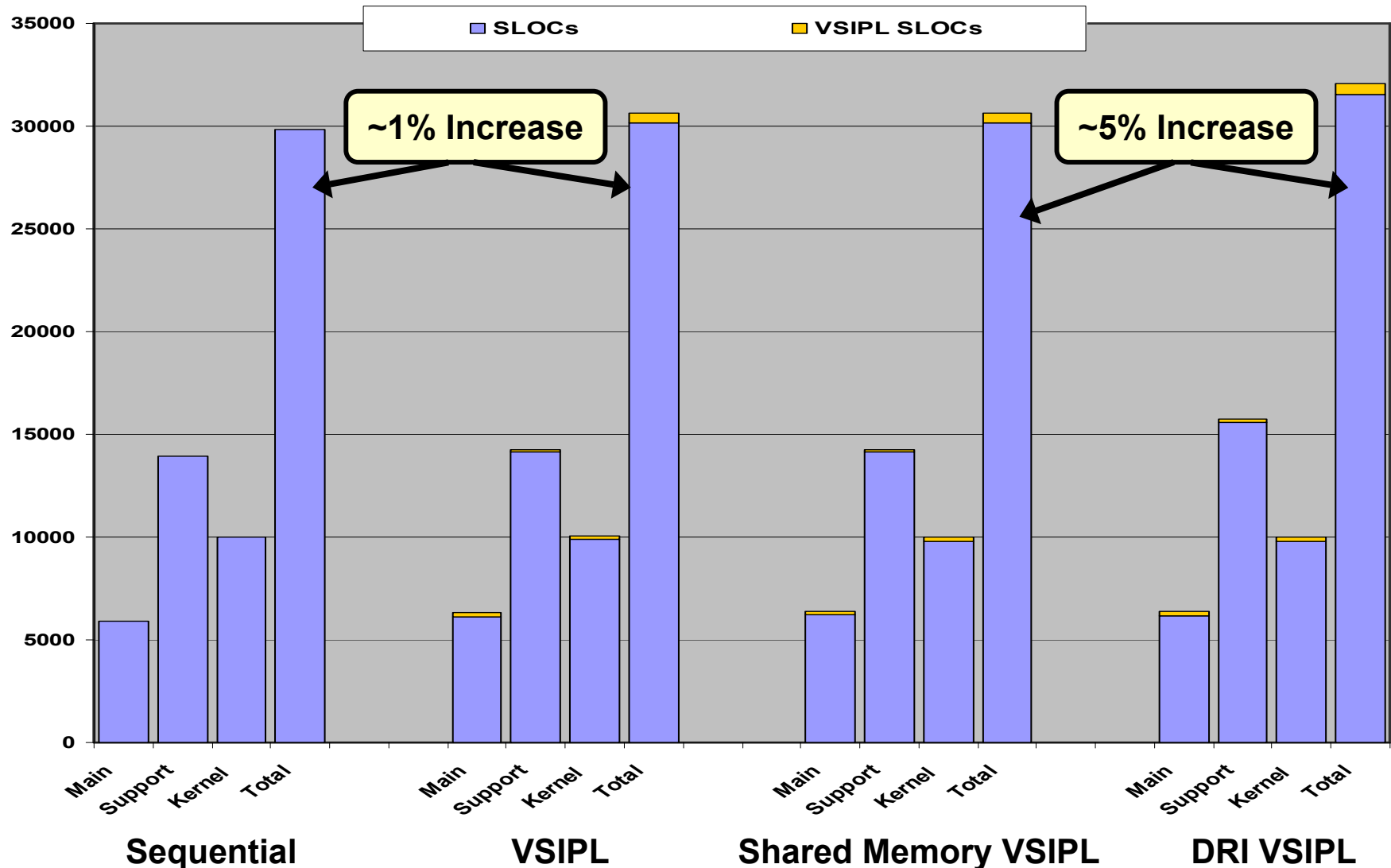
Embedded Multicomputers

- CSPI - 500MHz PPC7410 (vendor loan)
- Mercury - 500MHz PPC7410 (vendor loan)
- Sky - 333MHz PPC7400 (vendor loan)
- Sky - 500MHz PPC7410 (vendor loan)

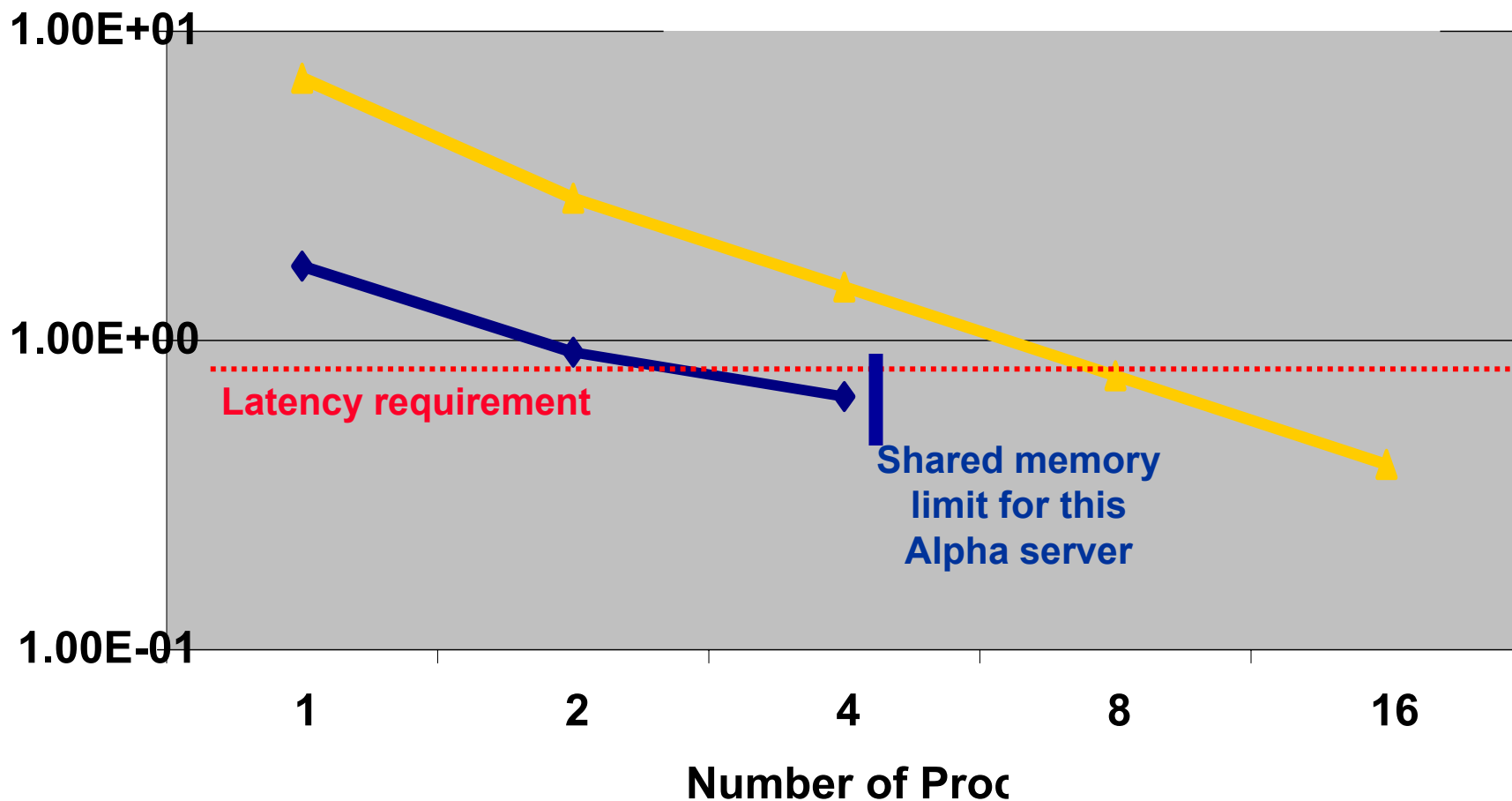
Mainstream Servers

- HP/COMPAQ ES40LP - 833-MHz Alpha ev6 (CIP hardware)
- HP/COMPAQ ES40 - 500-MHz Alpha ev6 (CIP hardware)
- SGI Origin 2000 - 250MHz R10k (CIP hardware)
- SGI Origin 3800 - 400MHz R12k (ARL MSRC)
- IBM 1.3GHz Power 4 (ARL MSRC)
- Generic LINUX Cluster

Portability: SLOC Comparison



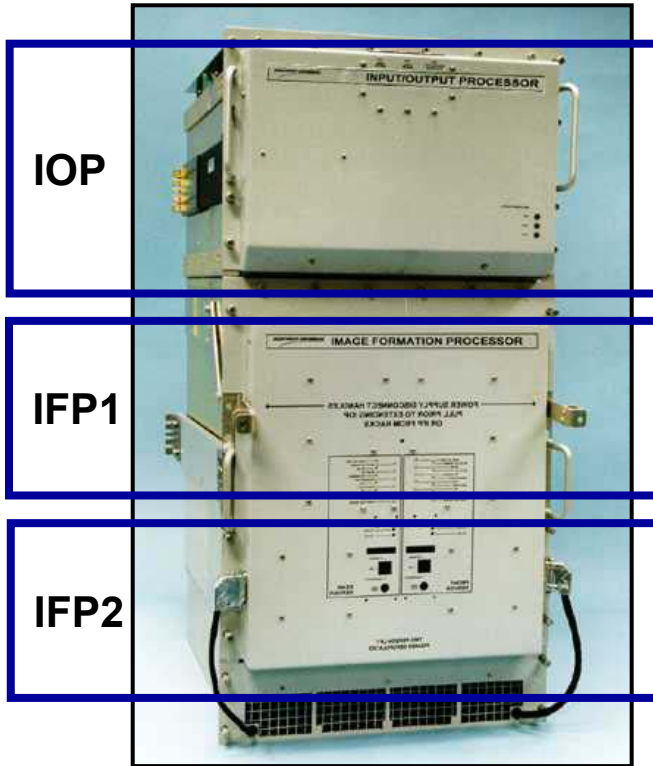
Shared Memory / CIP Server versus Distributed Memory / Embedded Vendor



Application can now exploit many more processors, embedded processors (3x form factor advantage) and Linux clusters (3x cost advantage)

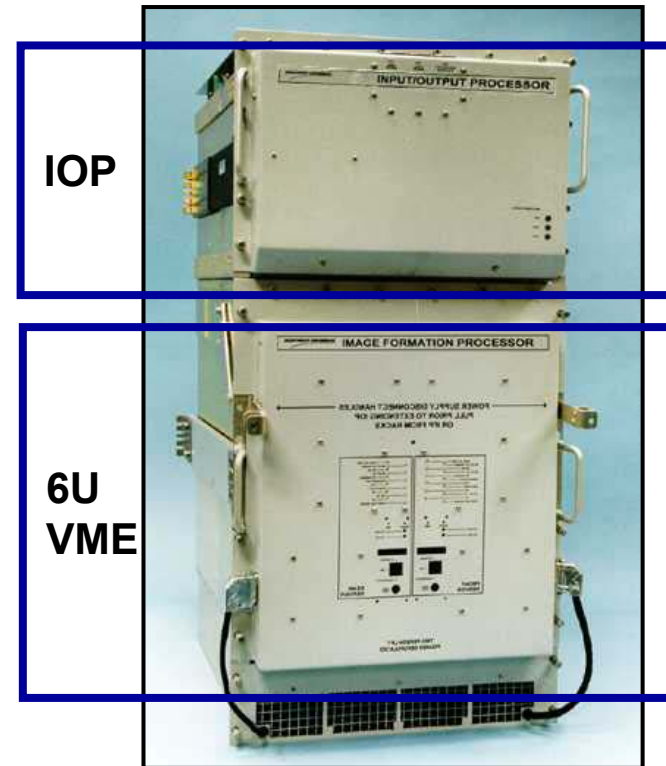
Form Factor Improvements

Current Configuration



- IOP: 6U VME chassis (9 slots potentially available)
- IFP: HP/COMPAQ ES40LP

Possible Configuration



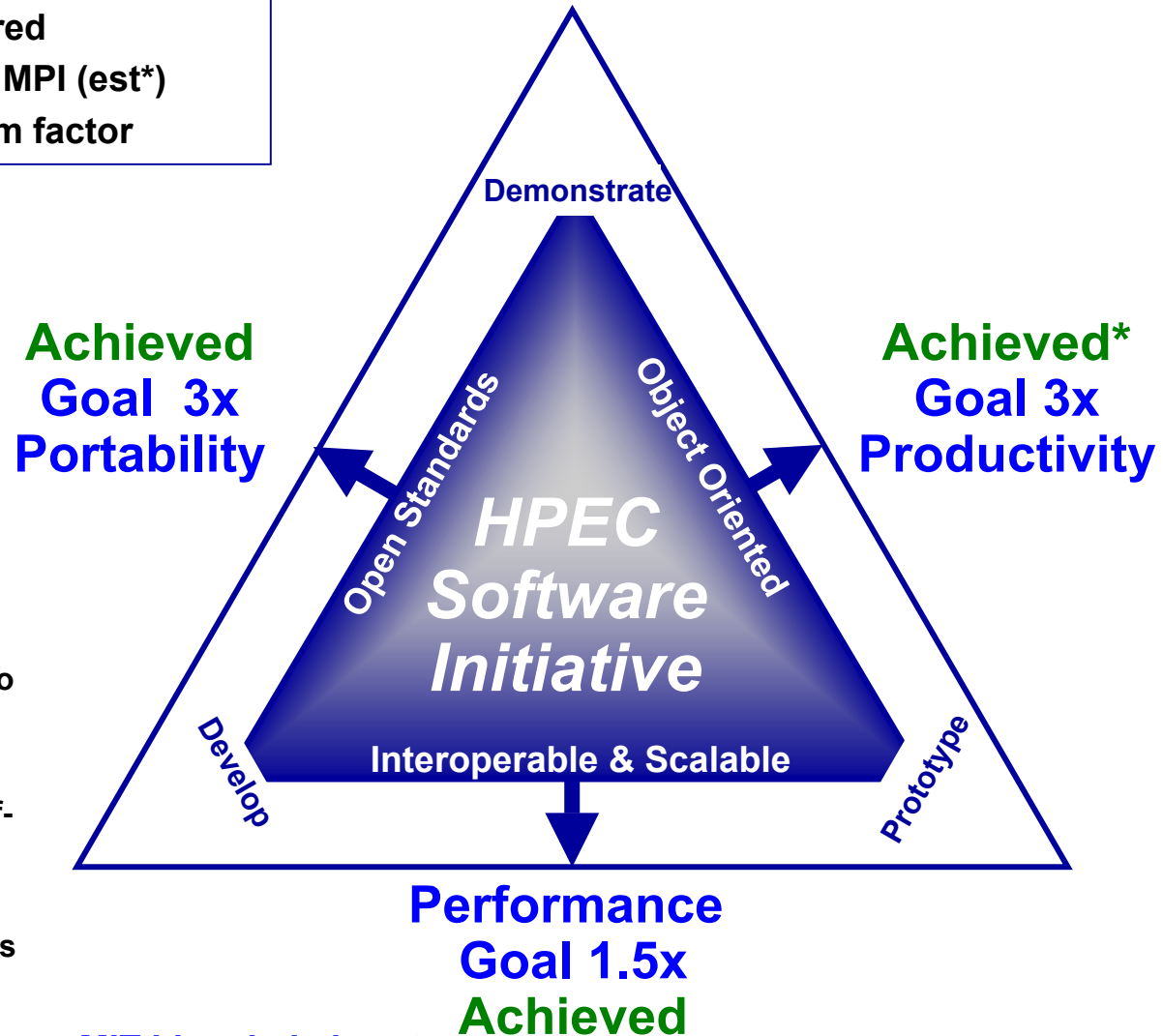
- IOP could support 2 G4 IFPs
 - form factor reduction (x2)
- 6U VME can support 5 G4 IFPs
 - processing capability increase (x2.5)



HPEC-SI Goals

1st Demo Achievements

Portability: zero code changes required
Productivity: DRI code 6x smaller vs MPI (est*)
Performance: 2x reduced cost or form factor



Portability: reduction in lines-of-code to change port/scale to new system

Productivity: reduction in overall lines-of-code

Performance: computation and communication benchmarks

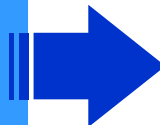


Outline



- Introduction
- Demonstration

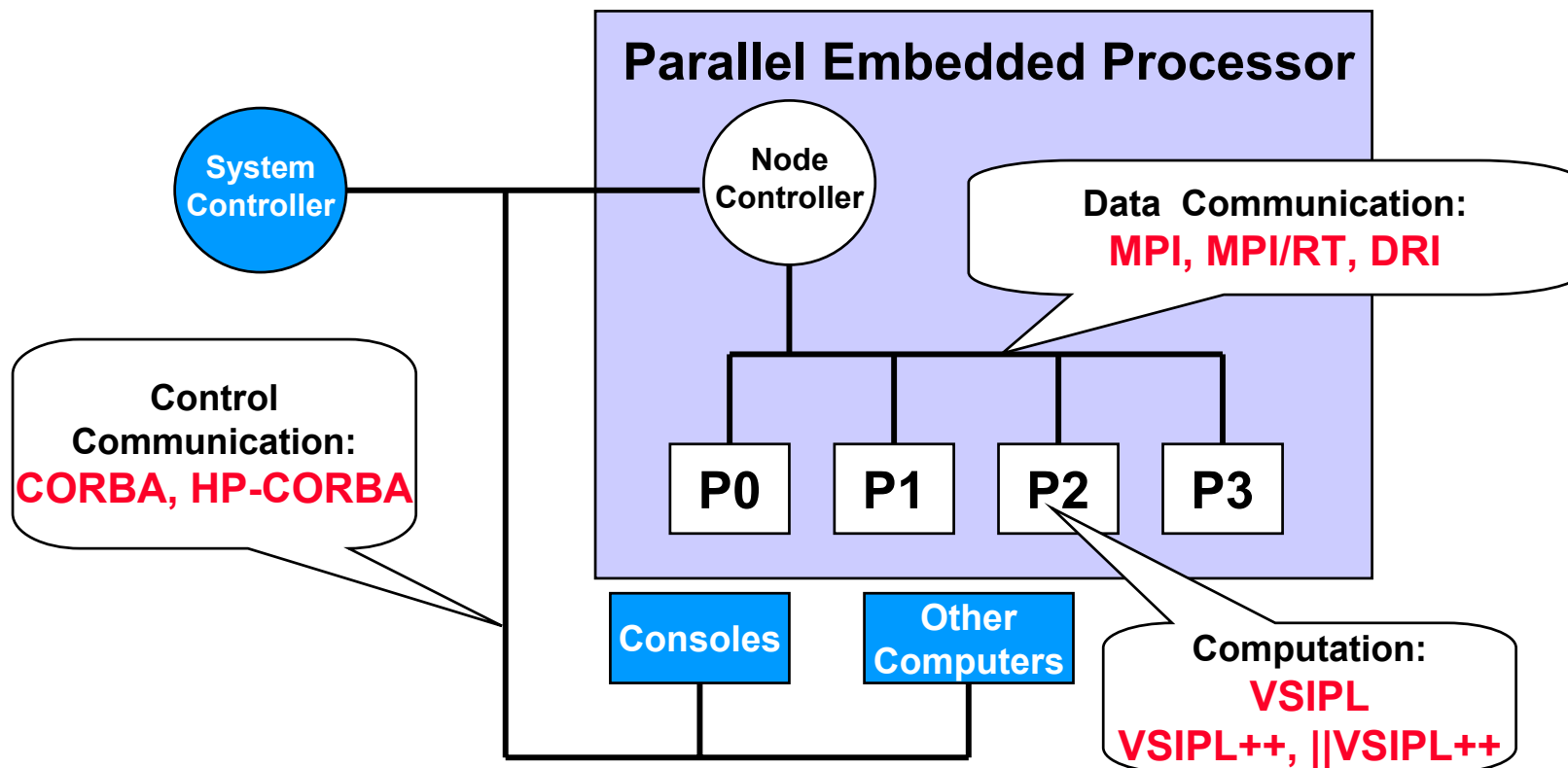
- **Development**



- *Object Oriented (VSIPL++)*
- *Parallel (||VSIPL++)*

- Applied Research
- Future Challenges
- Summary

Emergence of Component Standards



**HPEC Initiative - Builds on
completed research and existing
standards and libraries**

Definitions

VS IPL = Vector, Signal, and Image
Processing Library

||VS IPL++ = Parallel Object Oriented VS IPL

MPI = Message-passing interface

MPI/RT = MPI real-time

DRI = Data Re-org Interface

CORBA = Common Object Request Broker
Architecture

HP-CORBA = High Performance CORBA

VSIPL++ Productivity Examples

BLAS zherk Routine

- BLAS = Basic Linear Algebra Subprograms
- Hermitian matrix M: $\text{conj}(M) = M^t$
- zherk performs a rank-k update of Hermitian matrix C:

$$C \leftarrow \alpha * A * \text{conj}(A)^t + \beta * C$$

- VSIPL code

```
A = vsip_cmcreate_d(10,15,VSIP_ROW,MEM_NONE);
C = vsip_cmcreate_d(10,10,VSIP_ROW,MEM_NONE);
tmp = vsip_cmcreate_d(10,10,VSIP_ROW,MEM_NONE);
vsip_cmprodh_d(A,A,tmp); /* A*conj(A)^t */
vsip_rscmmul_d(alpha,tmp,tmp); /* alpha*A*conj(A)^t */
vsip_rscmmul_d(beta,C,C); /* beta*C */
vsip_cmadd_d(tmp,C,C); /* alpha*A*conj(A)^t + beta*C */
vsip_cblockdestroy(vsip_cmdestroy_d(tmp));
vsip_cblockdestroy(vsip_cmdestroy_d(C));
vsip_cblockdestroy(vsip_cmdestroy_d(A));
```

- VSIPL++ code (also parallel)

```
Matrix<complex<double>> > A(10,15);
Matrix<complex<double>> > C(10,10);
C = alpha * prodh(A,A) + beta * C;
```

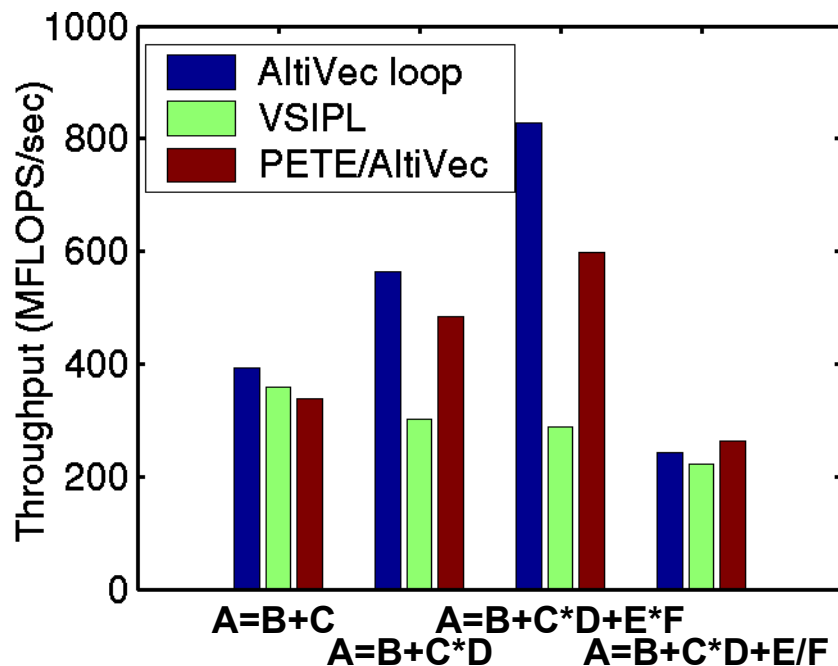
Sonar Example

- K-W Beamformer
- Converted C VSIPL to VSIPL++
- 2.5x less SLOCs

PVL PowerPC Altivec Experiments

Results

- Hand coded loop achieves good performance, but is problem specific and low level
- Optimized VSIPL performs well for simple expressions, worse for more complex expressions
- PETE style array operators perform almost as well as the hand-coded loop and are general, can be composed, and are high-level

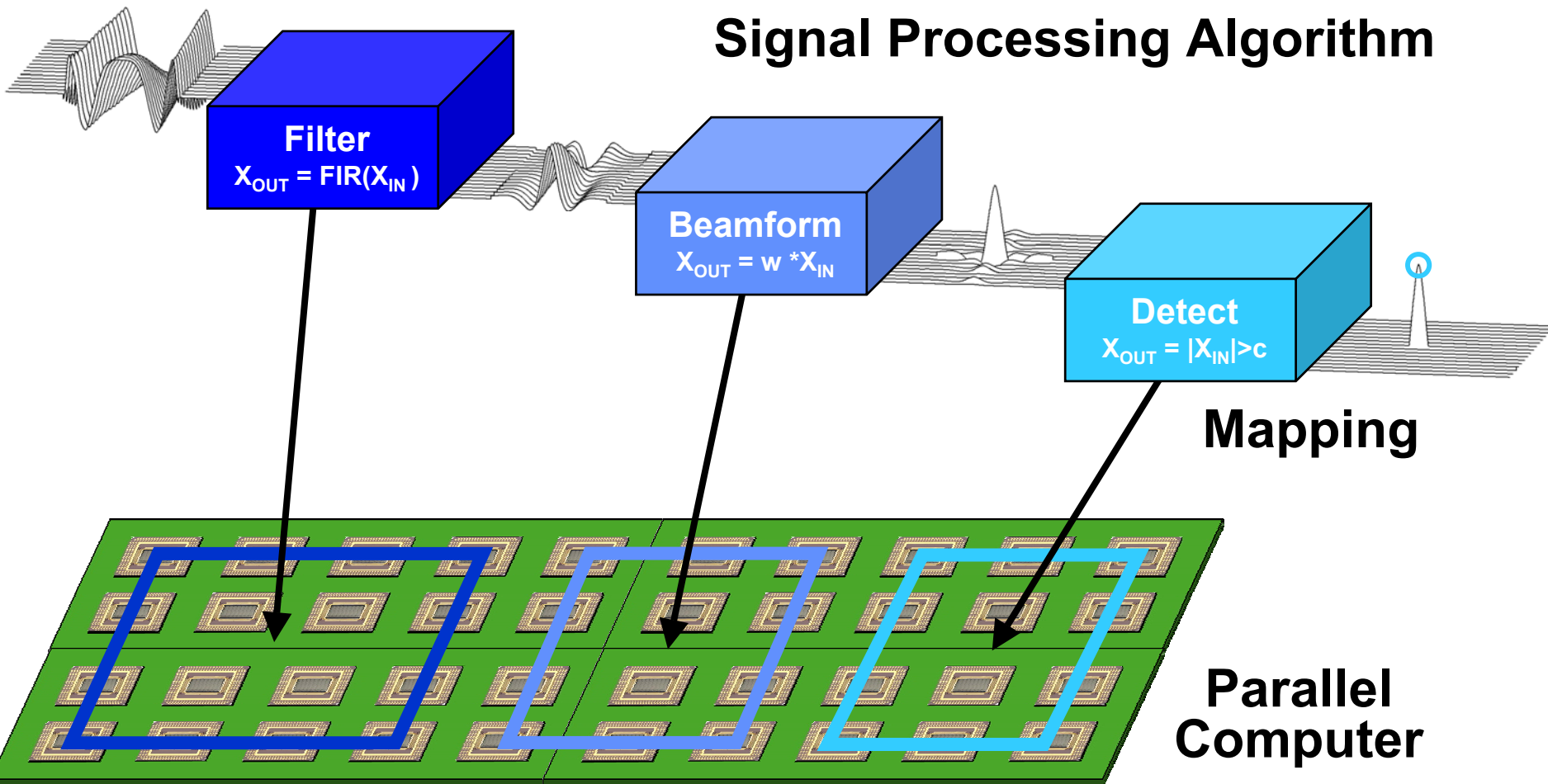


Software Technology

Altivec loop	VSIPL (vendor optimized)	PETE with Altivec
<ul style="list-style-type: none"> • C • For loop • Direct use of Altivec extensions • Assumes unit stride • Assumes vector alignment 	<ul style="list-style-type: none"> • C • Altivec aware VSIPPro Core Lite (www.mpi-softtech.com) • No multiply-add • Cannot assume unit stride • Cannot assume vector alignment 	<ul style="list-style-type: none"> • C++ • PETE operators • Indirect use of Altivec extensions • Assumes unit stride • Assumes vector alignment

Parallel Pipeline Mapping

Signal Processing Algorithm



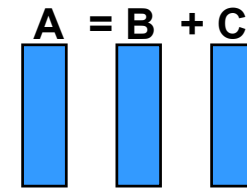
- Data Parallel within stages
- Task/Pipeline Parallel across stages

```
#include <Vector.h>
#include <AddPvl.h>

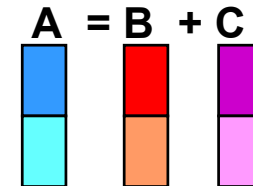
void addVectors(aMap, bMap, cMap) {
    Vector< Complex<Float> > a('a', aMap, LENGTH);
    Vector< Complex<Float> > b('b', bMap, LENGTH);
    Vector< Complex<Float> > c('c', cMap, LENGTH);

    b = 1;
    c = 2;
    a=b+c;
}
```

Single Processor Mapping



Multi Processor Mapping



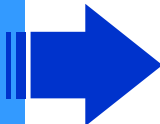
Lincoln Parallel Vector Library (PVL)

- Single processor and multi-processor code are the *same*
- *Maps* can be changed without changing software
- High level code is compact

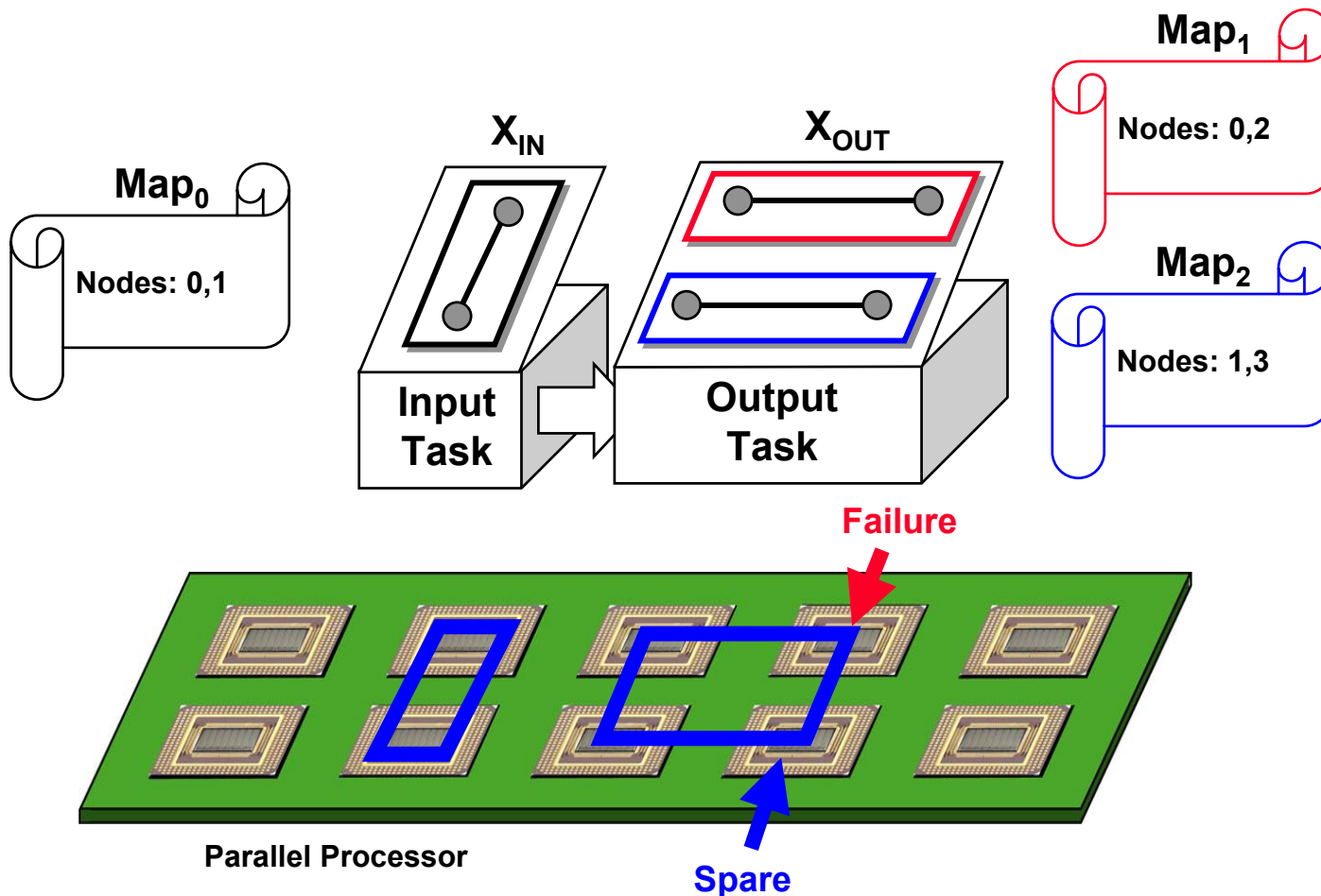


Outline



- Introduction
- Demonstration
- Development
- **Applied Research** 
 - *Fault Tolerance*
 - *Parallel Specification*
 - *Hybrid Architectures (see SBR)*
- Future Challenges
- Summary

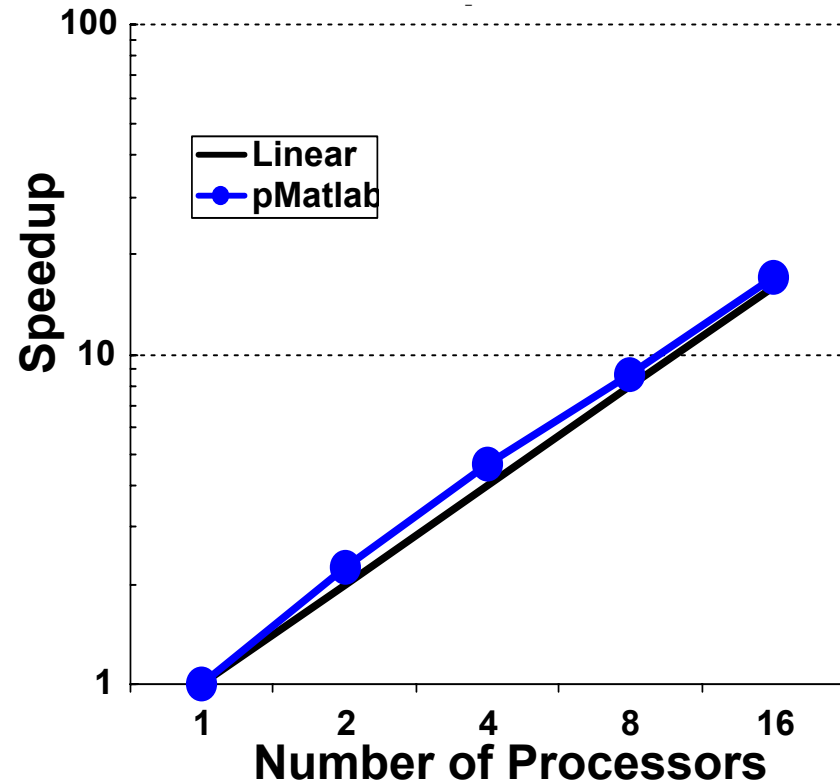
Dynamic Mapping for Fault Tolerance



- Switching processors is accomplished by switching maps
- No change to algorithm required
- Developing requirements for ||VS IPL++

Parallel Specification

Clutter Calculation (Linux Cluster)



```
% Initialize
pMATLAB_Init; Ncpus=comm_vars.comm_size;

% Map X to first half and Y to second half.
mapX=map([1 Ncpus/2],[],[1:Ncpus/2])
mapY=map([Ncpus/2 1],[],[Ncpus/2+1:Ncpus]);

% Create arrays.
X = complex(rand(N,M, mapX),rand(N,M, mapX));
Y = complex(zeros(N,M, mapY);

% Initialize coefficients
coefs = ...
weights = ...

% Parallel filter + corner turn.
Y(:, :) = conv2(coefs,X);
% Parallel matrix multiply.
Y(:, :) = weights*Y;

% Finalize pMATLAB and exit.
pMATLAB_Finalize; exit;
```

- Matlab is the main specification language for signal processing
- pMatlab allows parallel specifications using same mapping constructs being developed for ||VSIP++



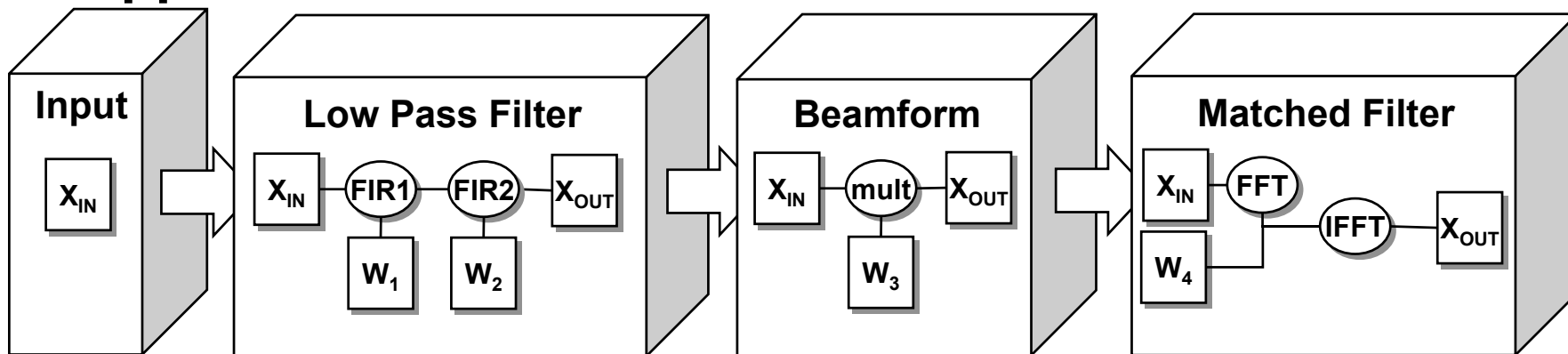
Outline



- Introduction
- Demonstration
- Development
- Applied Research
- **Future Challenges**
- Summary

Optimal Mapping of Complex Algorithms

Application



Different Optimal Maps



Workstation



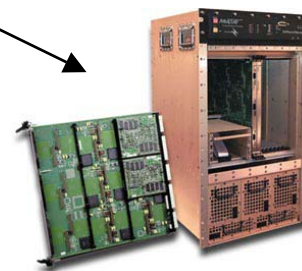
Intel Cluster



PowerPC Cluster



Embedded Board



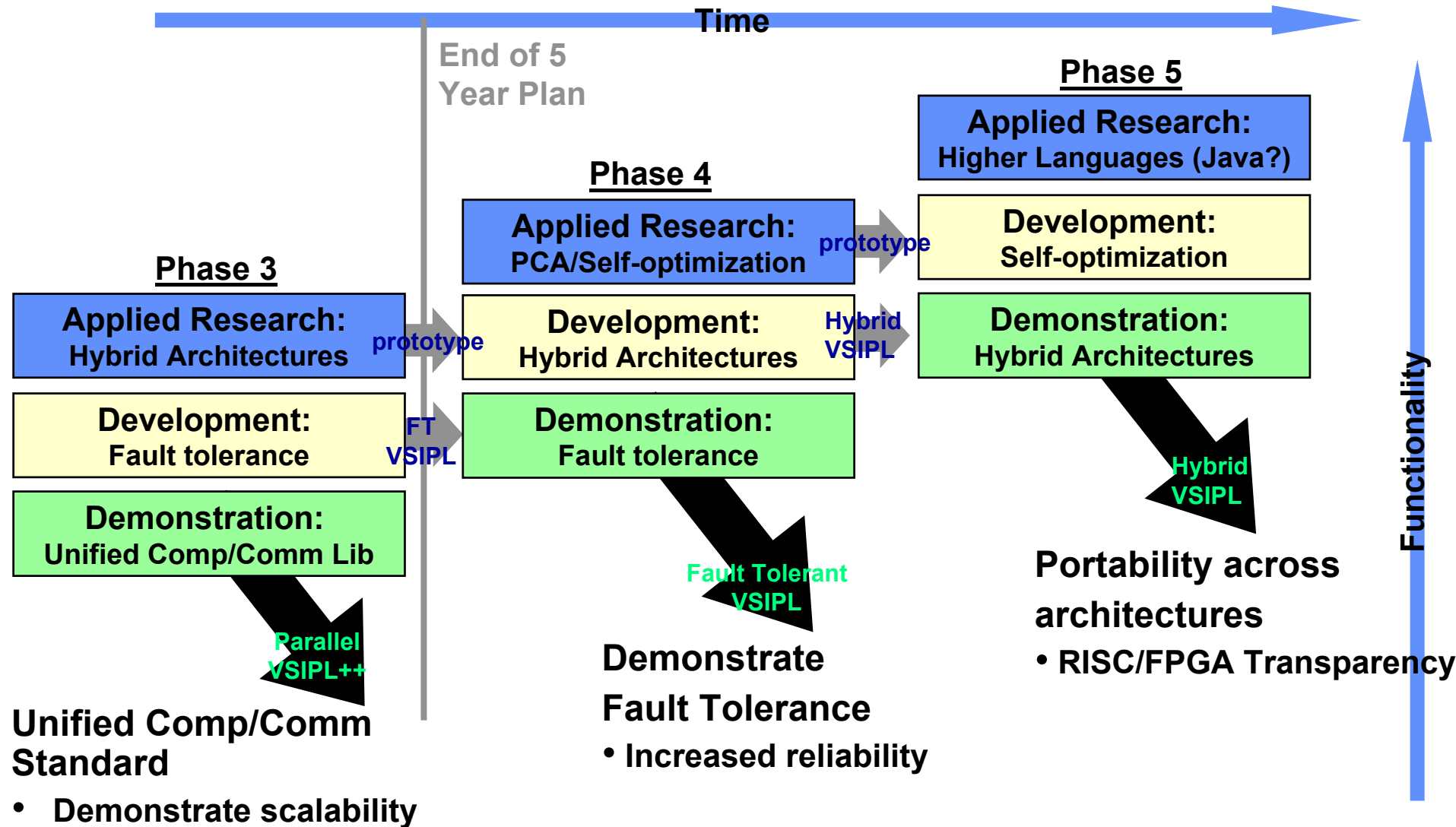
Embedded Multi-computer

Hardware

- Need to automate process of mapping algorithm to hardware



HPEC-SI Future Challenges





Summary



- **HPEC-SI Program on track toward changing software practice in DoD HPEC Signal and Image Processing**
 - Outside funding obtained for DoD program specific activities (on top of core HPEC-SI effort)
 - 1st Demo completed; 2nd selected
 - Worlds first parallel, object oriented standard
 - Applied research into task/pipeline parallelism; fault tolerance; parallel specification
- **Keys to success**
 - Program Office Support: 5 Year Time horizon better match to DoD program development
 - Quantitative goals for portability, productivity and performance
 - Engineering community support



Web Links

High Performance Embedded Computing Workshop

<http://www.ll.mit.edu/HPEC>

High Performance Embedded Computing Software Initiative

<http://www.hpec-si.org/>

Vector, Signal, and Image Processing Library

<http://www.vsipl.org/>

MPI Software Technologies, Inc.

<http://www.mpi-softtech.com/>

Data Reorganization Initiative

<http://www.data-re.org/>

CodeSourcery, LLC

<http://www.codesourcery.com/>

MatlabMPI

<http://www.ll.mit.edu/MatlabMPI>